IERG 4210

Web Programming and Security

Tutorial 6

ZHOU Yutong

Outline

- Tips for Phase 3
- Lecture review

Tips for Phase 3

- Using AJAX to get the price of the product, then calculate the total price and update the UI
- Store the pid and quantity of each product in Localstorage
- Restore the shopping list info through LocalStorage when page is reloaded.

Tips for Phase 3

- JavaScript: Dynamically update the shopping List
  - When click “Add to Cart” Button
  - When hover on the shopping List, a shopping list will expand
    - you can change the quantities of goods on the expended shopping list
  - When the page is reloaded, restore the shopping list from local storage.

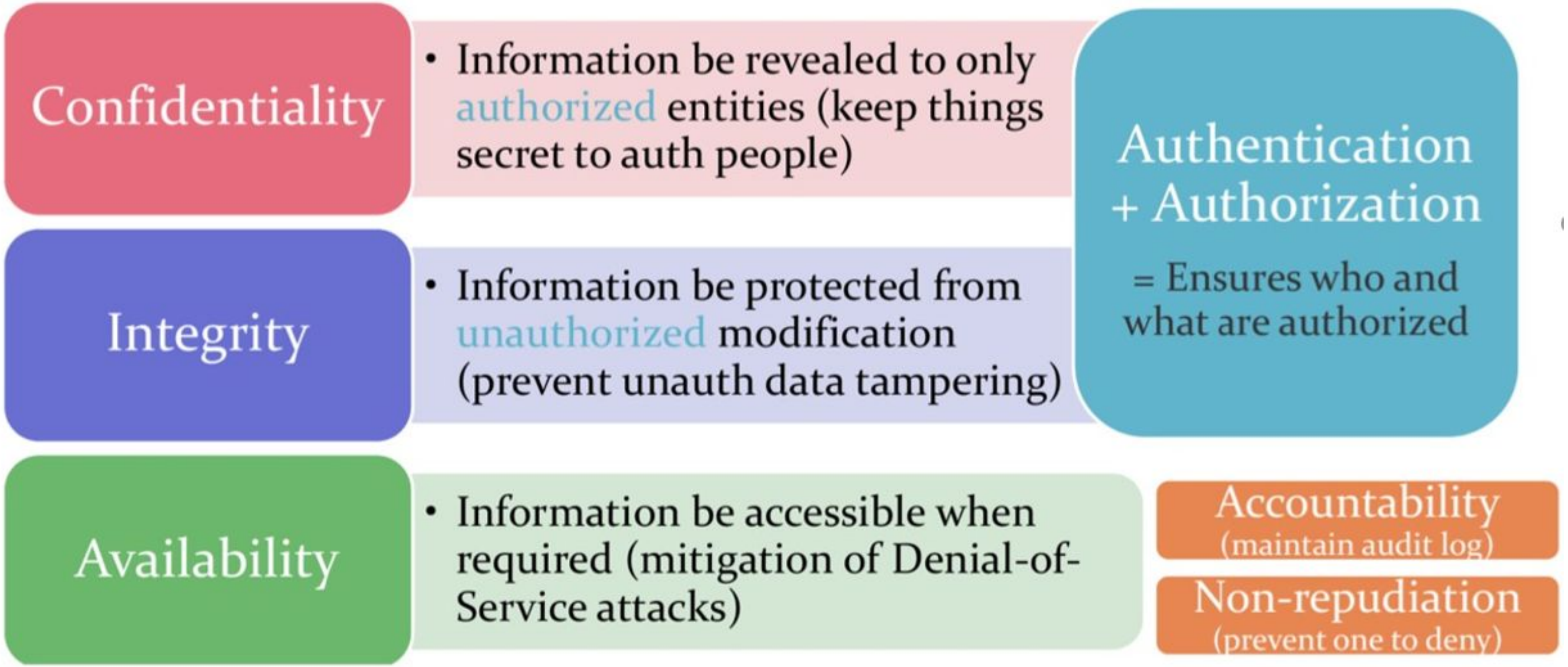
Basic Concepts of Web

- Web Architecture
  - HTTP, URL, etc.
- Web Development Languages
  - HTML, CSS, JavaScript, PHP, etc.
- Web Development Components
  - User Interface Design
    - Both Client and Server Side
  - Forms Handling
    - Both Client and Server Side
  - Web and Database Server Management
  - Session Management & Authentication

Basic Concepts of Internet

- Internet Components
  - URL: URL is a string that references an Internet resource.
  - Domain Name: Domain Name System (DNS) server resolves domain name to IP addresses for ease memorizing, or vise versa
  - IP Address: Address is a numerical address that references a device connecting to a computer network using the Internet Protocol.
  - World Wide Web: is the point-and-click system of navigating through information shared over the Internet by using hypertext

Information Security Goals



Confidentiality

- Information be revealed to only authorized entities (keep things secret to auth people)

Integrity

- Information be protected from unauthorized modification (prevent unauth data tampering)

Availability

- Information be accessible when required (mitigation of Denial-of-Service attacks)

Authentication + Authorization

= Ensures who and what are authorized

Accountability
(maintain audit log)

Non-repudiation
(prevent one to deny)

Secure Design Principle

- Securing the Weakest Link
- Secure Failure
- Defense-in-Depth
- Least-privilege
- Compartmentalization / Separation of Privilege
- Simplicity
- Promote Privacy
- Don’t extend trust easily

Client-side UI

- Structure and Content –HTML
- Presentation - Cascading Style Sheet (CSS)
- Behavior - JavaScript (JS)
  - An Object-Oriented Scripting Language
    - Dynamic Typing - Variable Types are generally dynamic
    - Interpreted Language - Just-In-Time (JIT) Compilation at browsers
    - Syntax - Similar to Java
- Data Object Model (DOM)
  - Browsers will parse a Web page file and build a tree-like data structure for it
  - Every <tag> corresponds to a Node Object, including CSS, JavaScript

JavaScript Events

- An element generates events that reflect its current status, which can be registered with event listening callback functions that respond accordingly.
- Asynchronous - Events are fired out of order
- Non-threaded - Events get queued and fired one at a time
- Some common types:
  - Mouse: click, mouseover, mouseout, dragstart*
  - Keyboard: keydown, keypress, keyup
  - TouchScreen: touchstart*, touchmove*, touchend*
  - Form/Input/Select: submit, change, focus
  - Un/Loading: load, beforeunload, error, readystatechange
  - Timer: setTimeout(), setTimeInterval()

Forms – Client-Side

- HTML Forms: Basic and Input Controls
- Client-Side Restrictions
    - The use of different form controls
    - Validations with HTML5
    - Validations with JavaScript
- Form Submission Approaches
    - Traditional Form Submission
    - Programmatic Form Submission
    - AJAX Form Submission

Forms – Server-Side

- Request Methods: Get vs. POST
- PHP, a server-side Scripting language:
  - Basics
  - C-like syntax with a few syntactic differences
  - Block-level Scoping for variables
- Form / Request Handling with PHP:
  - Input - Sanitizations and Validations
    - Code at client-side (for user experience enhancement)
    - Code at server-side (for security enforcement)
    - Security Best Practice (for input validation)

Forms – Server-Side

- Request Methods: Get vs. POST
- PHP, a server-side Scripting language:
  - Basics
  - C-like syntax with a few syntactic differences
  - Block-level Scoping for variables
- Form / Request Handling with PHP:
  - Input - Sanitizations and Validations
  - Process - Database Manipulation
    - SQL Languages (e.g., SELECT *)
    - DB Manipulations with PHP Data Objects (PDO)
  - Output - HTML vs. JSON

Forms – Server-Side

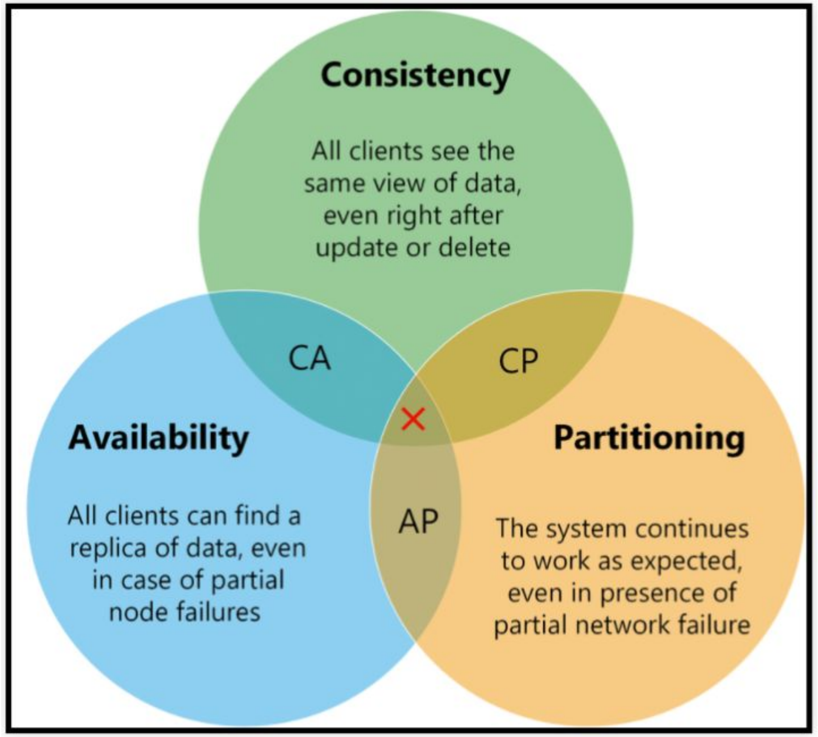
- Request Methods: Get vs. POST
- PHP, a server-side Scripting language:
  - Basics
  - C-like syntax with a few syntactic differences
  - Block-level Scoping for variables
- Form / Request Handling with PHP:
  - Input - Sanitizations and Validations
  - Process - Database Manipulation
  - Output – HTML vs. JSON
    - Advantages of using JSON when compared to HTML
      - Minimize bandwidth needed
      - JSON parsing is stunning fast as the format itself is JS
      - Loose coupling: PHP - data-intensive processing; JS - UI handling

Web & Database Servers

- Web Server on the Cloud
  - Quick Introduction to the Cloud
  - Architecture and Designs
- Database (DB) (Storage) Servers
  - Quick introduction to Database storage
    - Database vs. Cache
  - Relational Database (MySQL, SQLite)
  - NoSQL
  - Quick introduction to in-memory cache (redis)
  - CAP Theorem

Web & Database Server (CAP Theorem)

- Core Requirements of Distributed Systems
- Trilemma, you can only choose two (and relax the remaining)
  - C&A: Traditional Relational Database Management System
  - C&P: Redis
  - A&P: CouchDB


Consistency
All clients see the same view of data, even right after update or delete
CA
CP
×
Availability
Partitioning
All clients can find a replica of data, even in case of partial node failures
AP
The system continues to work as expected, even in presence of partial network failure

Web & Database Server

- Web Server on the Cloud
  - Quick Introduction to the Cloud
  - Architecture and Designs
- Database (DB) (Storage) Servers
- Database Integrity
  - concepts
    - Entity Integrity: every record (row) in a table is unique
    - Referential Integrity: data are consistent across multiple tables
    - Column Integrity: data of the same column have the same “type”
    - Other User-Defined Integrity: any special requests over the data
  - Two styles
    - Static: define some static constraints when creating the table
    - Dynamic: define some logic conditions or code that would be executed to perform the integrity check

Web & Database Server

- Web Server on the Cloud
  - Quick Introduction to the Cloud
  - Architecture and Designs
- Database (DB) (Storage) Servers
- Database Integrity
- Constraints to Achieve Integrity Goals
  - NOT NULL: Value must be defined
  - UNIQUE
  - Primary Key: unique + not-null
  - Foreign Key: to prevent illegal data

Authentication & Authorization

- Session Management
  - HTTP: from Stateless to Stateful
  - Session Maintenance: Cookie, HTML5 LocalStorage
    - Problems of Using Cookies
    - Cookie Integrity and Authenticity
    - Cookie Same Origin Policies (Cookie SOP)
      - Cookie Origin := (isHTTPSOnly, domain, path)
      - HTML Origin := (protocol, domain, port)
  - Extension to Server-Side Session Storage
    - Using a file-based system (most traditional)
    - Using a DB system
    - Using in-memory cache
  - Why not both?

Authentication & Authorization

- Session Management
  - HTTP: from Stateless to Stateful
  - Session Maintenance: Cookie, HTML5 LocalStorage
  - Extension to Server-Side Session Storage
- Authentication & Authorization
  - Authentication vs. Authorization
  - Authentication using Cookie
    - Authenticate the token before admin operations
    - Authorization check before admin operations
  - Authentication using HTTP Auth
    - The standardized and traditional way to authenticate a user
    - Not favorable by commercial websites since it's not customizable
  - Authentication Attacks